Latke CodeView - 2 Cron

作者：ZephyrJung
原文链接：https://ld246.com/article/1478185042517
来源网站：链滴

org.b3log.latke.cron.Cron

```
import org.b3log.latke.servlet.HTTPRequestMethod;
import org.b3log.latke.urlfetch.HTTPRequest;
import org.b3log.latke.urlfetch.URLFetchService;
import org.b3log.latke.urlfetch.URLFetchServiceFactory;
public final class Cron extends TimerTask {
    public static final int TEN = 10;
    public static final int SIXTY = 60;
    public static final int THOUSAND = 1000;
    private String url;
    private String description;
    private String schedule;
    private long period;
    public Cron(final String url, final String description, final String schedule) {
        this.url = url;
        this.description = description;
        this.schedule = schedule;
        parse(schedule);
    }
    public void run() {
        final URLFetchService urlFetchService = URLFetchServiceFactory.getURLFetchService();
        final HTTPRequest request = new HTTPRequest();
        try {
            request.setURL(new URL(url));
            request.setRequestMethod(HTTPRequestMethod.GET);
            urlFetchService.fetchAsync(request);
        } catch (final Exception e) {
        }
    }
    private void parse(final String schedule) {
        final int num = Integer.valueOf(StringUtils.substringBetween(schedule, " ", " "));
        final String timeUnit = StringUtils.substringAfterLast(schedule, " ");
        if ("hours".equals(timeUnit)) {
            period = num * SIXTY * SIXTY * THOUSAND;
        } else if ("minutes".equals(timeUnit)) {
            period = num * SIXTY * THOUSAND;
        } else if ("seconds".equals(timeUnit)) {
            period = num * THOUSAND;
        }
    }
    public long getPeriod() {
        return period;
    }
    public String getDescription() {
        return description;
    }
    public String getSchedule() {
        return schedule;
    }
    public String getURL() {
        return url;
    }
```

```
    public void setURL(final String url) {
        this.url = url;
    }
}
```

- TimerTask是实现了Runnable接口的类
- 主要有三个方法run，cancel， **scheduledExecutionTime**，用以启动，取消任务以及返回最后次执行时间（应该是这个意思吧？不确定）
- Cron继承了这个类，并添加了一些其他方法如解析时间定义

org.b3log.latke.cron.CronService

```
import org.b3log.latke.Latkes;
import org.b3log.latke.RuntimeEnv;
import org.b3log.latke.logging.Level;
import org.b3log.latke.logging.Logger;
import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.NodeList;
public final class CronService {
    private static final List<Cron> CRONS = new ArrayList<Cron>();
    private static final List<Timer> TIMERS = new ArrayList<Timer>();
    public static void start() {
        shutdown();
        final RuntimeEnv runtimeEnv = Latkes.getRuntimeEnv();
        try {
            switch (runtimeEnv) {
                case LOCAL:
                    loadCronXML();
                    for (final Cron cron : CRONS) {
                        cron.setURL(Latkes.getServer() + Latkes.getContextPath() + cron.getURL());
                        final Timer timer = new Timer();
                        TIMERS.add(timer);
                        timer.scheduleAtFixedRate(cron, Cron.TEN * Cron.THOUSAND, cron.getPeriod()
;
                    }
                    break;
                default:
                    throw new RuntimeException("Latke runs in the hell.... Please set the enviornment
correctly");
            }
        } catch (final Exception e) {
            throw new IllegalStateException(e);
        }
    }
    public static void shutdown() {
        CRONS.clear();
        for (final Timer timer : TIMERS) {
            timer.cancel();
        }
        TIMERS.clear();
    }
    private static void loadCronXML() {
```

```
        final File cronXML = Latkes.getWebFile("/WEB-INF/cron.xml");
        if (null == cronXML || !cronXML.exists()) {
            return;
        }
        final DocumentBuilderFactory documentBuilderFactory = DocumentBuilderFactory.newI
nstance();
        try {
            final DocumentBuilder documentBuilder = documentBuilderFactory.newDocumentBui
der();
            final Document document = documentBuilder.parse(cronXML);
            final Element root = document.getDocumentElement();
            root.normalize();
            final NodeList crons = root.getElementsByTagName("cron");
            for (int i = 0; i < crons.getLength(); i++) {
                final Element cronElement = (Element) crons.item(i);
                final Element urlElement = (Element) cronElement.getElementsByTagName("url").it
m(0);
                final Element descriptionElement = (Element) cronElement.getElementsByTagName(
description").item(0);
                final Element scheduleElement = (Element) cronElement.getElementsByTagName("s
hedule").item(0);
                final String url = urlElement.getTextContent();
                final String description = descriptionElement.getTextContent();
                final String schedule = scheduleElement.getTextContent();
                CRONS.add(new Cron(url, description, schedule));
            }
        } catch (final Exception e) {
            throw new RuntimeException(e);
        }
    }
    private CronService() {
    }
}
```

- 我发现D大喜欢先将对象添加到集合中，再给对象赋值，与我习惯正好相反
- Timer，主要有cacel,purge,shedule,scheduleAtFixedRate几个方法，分别用于取消定时任务，清定时任务，计划执行，以固定频率计划执行
- Timer是定时任务执行器，Cron因为继承了TimerTask故而可以作为参数传进去并启动执行（按照定频率）
- **一个Timer对应一个TimerTask，不知道我这样想对不对**